

(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号

特開2003-22621

(P2003-22621A)

(43)公開日 平成15年1月24日(2003.1.24)

(51)Int.Cl. <sup>7</sup>	識別記号	F I	キーワード*(参考)
G 1 1 B 20/12	1 0 3	G 1 1 B 20/12	5 C 0 5 3
20/10		20/10	1 0 3 5 D 0 4 4
27/00		27/00	G 5 D 1 1 0
H 0 4 N 5/92		H 0 4 N 5/92	A
			H
審査請求 未請求 請求項の数8 O L (全 24 頁)			

(21)出願番号 特願2001-207244(P2001-207244)

(22)出願日 平成13年7月9日(2001.7.9)

(71)出願人 000005049

シャープ株式会社

大阪府大阪市阿倍野区長池町22番22号

(72)発明者 木山 次郎

大阪府大阪市阿倍野区長池町22番22号 シ

ャープ株式会社内

(72)発明者 岩野 裕利

大阪府大阪市阿倍野区長池町22番22号 シ

ャープ株式会社内

(74)代理人 100102277

弁理士 佐々木 晴康 (外2名)

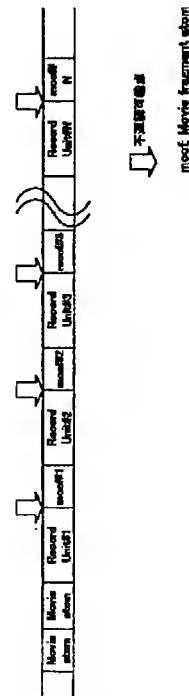
最終頁に続く

(54)【発明の名称】 データ記録方法、データ変更方法及びその装置

(57)【要約】

【課題】 AVストリームに多重化した管理情報の更新を容易にする。

【解決手段】 映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニット(Record Unit)と、前記第1のユニットを管理する管理情報(Movie fragment atom)とを、記録媒体に記録するデータ記録方法であって、前記第1のユニットと前記管理情報とを、前記記録媒体上で近傍に配置するものである。



## 【特許請求の範囲】

【請求項1】 映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のユニットを管理する管理情報とを、記録媒体に記録するデータ記録方法であって、

前記第1のユニットと前記管理情報とを前記記録媒体上で近傍に配置することを特徴とするデータ記録方法。

【請求項2】 映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のデータを管理する第1の管理情報と、前記第2のデータを管理する第2の管理情報とを、記録媒体に記録するデータ記録方法であって、

前記第2の管理情報と前記第1の管理情報とを前記記録媒体上で分離して配置するとともに、1個以上の前記第2の管理情報を互いに前記記録媒体上で近傍に配置することを特徴とするデータ記録方法。

【請求項3】 映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のユニットを管理する管理情報とが、近傍に配置されて記録された記録媒体に対し、前記第2のデータと前記管理情報とを書き換えるデータ変更方法であって、前記第2のデータと前記管理情報とを連続的に書き換えることを特徴とするデータ変更方法。

【請求項4】 映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のデータを管理する第1の管理情報とが記録され、さらに、前記第2の管理情報が1個以上互いに近傍に配置されて記録された記録媒体に対し、前記第2のデータと前記第2の管理情報とを書き換えるデータ変更方法であって、前記第2の管理情報を連続して書き換えることを特徴とするデータ変更方法。

【請求項5】 映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のユニットを管理する管理情報とを、記録媒体に記録するデータ記録装置であって、前記管理情報を前記記録媒体上で前記第1のユニットの近傍に配置して記録する手段を備えたことを特徴とするデータ記録装置。

【請求項6】 映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のデータを管理する第1の管理情報と、前記第2のデータを管理する第2の管理情報とを、記録媒体に記録するデータ記録装置であって、

前記第2の管理情報と前記第1の管理情報とを前記記録

媒体上で分離して配置するとともに、1個以上の前記第2の管理情報を互いに前記記録媒体上で近傍に配置して記録する手段を備えたことを特徴とするデータ記録装置。

【請求項7】 映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のユニットを管理する管理情報とが、近傍に配置されて記録された記録媒体に対し、前記第2のデータと前記管理情報とを書き換えるデータ変更装置であって、

前記管理情報を前記第2のデータと連続的に書き換える手段を備えたことを特徴とするデータ変更装置。

【請求項8】 映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のデータを管理する第1の管理情報とが記録され、さらに、前記第2の管理情報が1個以上互いに近傍に配置されて記録された記録媒体に対し、前記第2のデータと前記第2の管理情報とを書き換えるデータ変更装置であって、

前記第2の管理情報を連続して書き換える手段を備えたことを特徴とするデータ変更装置。

## 【発明の詳細な説明】

## 【0001】

【発明の属する技術分野】本発明は、ハードディスク、光ディスク等のランダムアクセス可能な記録媒体に対して、映像データ、音声データを記録・変更するデータ記録方法、データ変更方法及びその装置に関するものである。

## 【0002】

【従来の技術】ディスクメディアを用いたビデオのデジタル記録再生装置が普及しつつある。ディスクメディアではテープメディアと異なり、任意の箇所へのアクセスが短時間で可能であり、その性質を利用して、データをコピーすることなく1枚のディスク上で管理情報の書換のみにより編集を行う非破壊編集あるいはノンリニア編集と呼ばれる機能や、記録済みのデータを上書きすることなく、削除した後の分散した空き領域にまたがるビデオを記録する機能を実現することができる。

【0003】ディスクメディアを用いたビデオ録画装置においては、録画時に管理情報を最後にまとめて記録することが一般的である。しかし、録画中に不意に電源が切れたりして管理情報を記録することができなかった場合、それまでに録画したデータを管理することができなくなってしまう、すなわち再生不能になってしまうという問題がある。

【0004】この問題に対応するため、例えばMotion JPEG 2000では、Fragmented movieという概念が導入されている。ここでは、Fragmented movieについてその概略を説明する。Fragmented movieを含むMotion JPEG 2000 ファイルの典型的な構成を図32に示す。

【0005】先頭にそのファイル全体に共通する情報を管理するMovie atomが配置され、その後、部分AVストリームデータ (Movie fragmentと呼ぶ) を格納するMovie data atomと、そのMovie fragmentを管理するMovie fragment atomとが交互に配置される。

【0006】録画時にはこの順番で記録を行なっていくことにより、仮に録画中に電源が切れた場合でも直前に記録したMovie fragment atomと、それが管理するMovie fragmentまではディスクに残っており、後で再生することが可能となる。

【0007】

【発明が解決しようとする課題】ディスクメディアにおいても、テープメディアと同様、アフレコ機能が要求される。アフレコ機能に対応したAVストリームとして、図33に示すように、アフレコ時にデータを格納するための領域を多重化したものが考えられる。

【0008】アフレコしたデータに関してもユーザの資産であり、管理情報のバックアップにより極力保護することが望まれる。しかしながら、上述のMotion JPEG2000規格においては、このようなストリームに関する管理情報の構成がない。

【0009】本発明は、上記課題に鑑みてなされたものであり、アフレコデータに関する管理情報の保護を可能にするデータ記録方法、データ変更方法を提供することを目的とする。

【0010】

【課題を解決するための手段】本願の第1の発明は、映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のユニットを管理する管理情報とを、記録媒体に記録するデータ記録方法であって、前記第1のユニットと前記管理情報とを前記記録媒体上で近傍に配置することを特徴とする。

【0011】本願の第2の発明は、映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のデータを管理する第1の管理情報と、前記第2のデータを管理する第2の管理情報とを、記録媒体に記録するデータ記録方法であって、前記第2の管理情報と前記第1の管理情報とを前記記録媒体上で分離して配置するとともに、1個以上の前記第2の管理情報を互いに前記記録媒体上で近傍に配置することを特徴とする。

【0012】本願の第3の発明は、映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のユニットを管理する管理情報とが、近傍に配置されて記録された記録媒体に対し、前記第2のデータと前記管理情報とを書き換えるデータ変更方法であって、前記第2のデータと前記管理情報とを連続的に書き換えることを特徴とする。

【0013】本願の第4の発明は、映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のデータを管理する第1の管理情報とが記録され、さらに、前記第2の管理情報が1個以上互いに近傍に配置されて記録された記録媒体に対し、前記第2のデータと前記第2の管理情報とを書き換えるデータ変更方法であって、前記第2の管理情報を連続して書き換えることを特徴とする。

10 【0014】本願の第5の発明は、映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のユニットを管理する管理情報とを、記録媒体に記録するデータ記録装置であって、前記管理情報を前記記録媒体上で前記第1のユニットの近傍に配置して記録する手段を備えたことを特徴とする。

【0015】本願の第6の発明は、映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のデータを管理する第1の管理情報と、前記第2のデータを管理する第2の管理情報とを、記録媒体に記録するデータ記録装置であって、前記第2の管理情報と前記第1の管理情報とを前記記録媒体上で分離して配置するとともに、1個以上の前記第2の管理情報を互いに前記記録媒体上で近傍に配置して記録する手段を備えたことを特徴とする。

【0016】本願の第7の発明は、映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のユニットを管理する管理情報とが、近傍に配置されて記録された記録媒体に対し、前記第2のデータと前記管理情報とを書き換えるデータ変更装置であって、前記管理情報を前記第2のデータと連続的に書き換える手段を備えたことを特徴とする。

【0017】本願の第8の発明は、映像又は音声からなる第1のデータと、前記第1のデータと同期して再生される第2のデータとを格納する第1のユニットと、前記第1のデータを管理する第1の管理情報とが記録され、さらに、前記第2の管理情報が1個以上互いに近傍に配置されて記録された記録媒体に対し、前記第2のデータと前記第2の管理情報とを書き換えるデータ変更装置であって、前記第2の管理情報を連続して書き換える手段を備えたことを特徴とする。

【0018】

【発明の実施の形態】以下、本発明の実施形態について、図面を参照しながら詳細に説明する。

【0019】＜システム構成＞図1は本実施形態において共通に用いる、アフレコ可能なビデオディスクレコーダの構成図である。この装置は、図1に示すように、バス100、ホストCPU101、RAM102、ROM103、ユーザインタ

フェース104、システムクロック105、光ディスク106、ピックアップ107、ECCデコーダ108、ECCエンコーダ109、再生用バッファ110、記録/アフレコ用バッファ111、デマルチプレクサ112、マルチプレクサ113、多重化用バッファ114、オーディオデコーダ115、ビデオデコーダ116、オーディオエンコーダ117、ビデオエンコーダ118、および図示しないカメラ、マイク、スピーカ、ディスプレイ等で構成される。

【0020】ホストCPU101は、バス100を通じてデマルチプレクサ112、マルチプレクサ113、ピックアップ107、また図示していないが、オーディオデコーダ115、ビデオデコーダ116、オーディオエンコーダ117、ビデオエンコーダ118との通信を行う。

【0021】再生時に、光ディスク106からピックアップ107を通じて読み出されたデータは、ECCデコーダ108によって誤り訂正され、再生用バッファ110に一旦蓄えられる。デマルチプレクサ112はオーディオデコーダ115、ビデオデコーダ116からのデータ送信要求に従い、再生用バッファ中のデータをその種別によって適当なデコーダに振り分ける。

【0022】一方、記録時に、オーディオエンコーダ117とビデオエンコーダ118によって圧縮符号化されたデータは、多重化用バッファ114に一旦送られ、マルチプレクサ113によってAV多重化され、記録/アフレコ用バッファ111に送られる。記録/アフレコ用バッファ111中のデータは、ECCエンコーダ109によって誤り訂正符号を付加され、ピックアップ107を通じて光ディスク106に記録される。

【0023】オーディオデータの符号化方式にはMPEG-1 Layer-IIを、ビデオデータの符号化方式にはMPEG-2をそれぞれ用いる。

【0024】光ディスク106は、外周から内周に向かって螺旋状に記録再生が行われる脱着可能な光ディスクとする。2048byteを1セクタとし、誤り訂正のため16セクタでECCブロックを構成する。ECCブロック中のデータを書き換える場合、そのデータが含まれるECCブロック全体を読み込み、誤り訂正を行って、対象のデータを書き換え、再び誤り訂正符号を付加し、ECCブロックを構成して、記録媒体に記録する必要がある。また、光ディスク106は、記録効率を上げるためZCAV（ゾーン角速度一定）を採用しており、記録領域は回転数の異なる複数のゾーンで構成される。

【0025】＜ファイルシステム＞光ディスク106上の各種情報を管理するためにファイルシステムを用いる。ファイルシステムには、パーソナルコンピュータ（PC）との相互運用を考慮してUDF（Universal Disk Format）を使用する。ファイルシステム上では、各種管理情報やAVストリームはファイルとして扱われる。

【0026】ユーザエリアは、2048byteの論理ブロック（セクタと一対一対応）で管理される。各ファイルは、

整数個のエクステンツ（連続した論理ブロック）で構成され、エクステンツ単位で分散して記録しても良い。空き領域は、Space Bitmapを用いて論理ブロック単位で管理される。

【0027】＜ファイルフォーマット＞AVストリーム管理のためのフォーマットとして、QuickTimeファイルフォーマットを用いる。QuickTimeファイルフォーマットとは、Apple社が開発したマルチメディアデータ管理用フォーマットであり、PCの世界で広く用いられている。

【0028】QuickTimeファイルフォーマットは、ビデオデータやオーディオデータ等（これらを総称してメディアデータとも呼ぶ）と管理情報とで構成される。両者を合わせてここでは、QuickTimeムービー（略してムービー）と呼ぶ。両者は同じファイル中に存在しても、別々のファイルに存在しても良い。

【0029】同じファイル中に存在する場合は、図2（a）に示すような構成をとる。各種情報はatomという共通の構造に格納される。管理情報はMovie atomという構造に格納され、AVストリームはMovie data atomという構造に格納される。尚、Movie atom中の管理情報には、メディアデータ中の任意の時間に対応するAVデータのファイル中での相対位置を導くためのテーブルや、メディアデータの属性情報や、後述する外部参照情報等が含まれている。

【0030】一方、管理情報とメディアデータを別々のファイルに格納した場合は、図2（b）に示すような構成をとる。管理情報はMovie atomという構造に格納されるが、AVストリームはatomには格納される必要はない。このとき、Movie atomはAVストリームを格納したファイルを「外部参照」している、という。

【0031】外部参照は、図2（c）に示すように、複数のAVストリームファイルに対して行うことが可能であり、この仕組みにより、AVストリーム自体を物理的に移動することなく、見かけ上編集を行ったように見せる、いわゆる「ノンリニア編集」「非破壊編集」が可能になる。

【0032】それでは、図3乃至図12を用いて、QuickTimeの管理情報のフォーマットについて説明する。まず、共通の情報格納フォーマットであるatomについて説明する。atomの先頭には、そのatomのサイズであるAtom size、そのatomの種別情報であるTypeが必ず存在する。Typeは4文字で区別され、例えばMovie atomでは'mov'、Movie data atomでは'mdat'となっている。

【0033】各atomは別のatomを含むことができる。すなわち、atom間には階層構造がある。Movie atomの構成を図3に示す。Movie header atomは、そのMovie atomが管理するムービーの全体的な属性を管理する。Track atomは、そのムービーに含まれるビデオやオーディオ等のトラックに関する情報を格納する。User data atomは、独自に定義可能なatomである。

【0034】Track atomの構成を図4に示す。Track header atomは、そのトラックの全体的な属性を管理する。Edit atomは、メディアデータのどの区間を、ムービーのどのタイミングで再生するかを管理する。Track reference atomは、別のトラックとの関係を管理する。Media atomは、実際のビデオやオーディオといったデータを管理する。

【0035】Track header atomの構成を図5に示す。ここでは、後での説明に必要なもののみについて説明する。Flagsは属性を示すフラグの集合である。代表的なものとして、Track enabledフラグがあり、このフラグが1であれば、そのトラックは再生され、0であれば再生されない。Layerはそのトラックの空間的な優先度を表しており、画像を表示するトラックが複数あれば、Layerの値が小さいトラックほど画像が前面に表示される。

【0036】Media atomの構成を図6に示す。Media header atomは、そのMedia atomの管理するメディアデータに関する全体的な属性等を管理する。Handler reference atomは、メディアデータをどのデコーダでデコードするかを示す情報を格納する。Media information atomは、ビデオやオーディオ等メディア固有の属性情報を管理する。

【0037】Media information atomの構成を図7に示す。Media information header atomは、ビデオやオーディオ等メディア固有の属性情報を管理する。Handler reference atomは、Media atomの項で説明した通りである。Data information atomは、そのQuickTimeムービーが参照するメディアデータを含むファイルの名前を管理するatomであるData reference atomを含む。Sample table atomは、データのサイズや再生時間等を管理している。

【0038】次に、Sample table atomについて説明するが、その前に、QuickTimeにおけるデータの管理方法について、図8を用いて説明する。QuickTimeでは、データの最小単位（例えばビデオフレーム）をサンプルと呼ぶ。個々のトラック毎に、サンプルには再生時間順に1から番号（サンプル番号）がついている。

【0039】また、QuickTimeフォーマットでは、個々のサンプルの再生時間長およびデータサイズを管理している。また、同一トラックに属するサンプルが再生時間順にファイル中で連続的に配置された領域をチャンクと呼ぶ。チャンクにも再生時間順に、1から番号がついている。

【0040】さらに、QuickTimeフォーマットでは、個々のチャンクのファイル先頭からのアドレスおよび個々のチャンクが含むサンプル数を管理している。これらの情報に基づき、任意の時間に対応するサンプルの位置を求めることが可能となっている。

【0041】Sample table atomの構成を図9に示す。Sample description atomは、個々のチャンクのデータフ

ォーマット（Data format）やサンプルが格納されているファイルのチャンクのインデックス等を管理する。Time-to-sample atomは、個々のサンプルの再生時間を管理する。

【0042】Sync sample atomは、個々のサンプルのうち、デコード開始可能なサンプルを管理する。Sample-to-chunk atomは、個々のチャンクに含まれるサンプル数を管理する。Sample size atomは、個々のサンプルのサイズを管理する。Chunk offset atomは、個々のチャンクのファイル先頭からのアドレスを管理する。

【0043】Edit atomは、図10に示すように、1個のEdit list atomを含む。Edit list atomはNumber of entriesで指定される個数分の、Track duration、Media time、Media rateの値の組（エン트리）を持つ。各エントリは、トラック上で連続的に再生される区間に対応し、そのトラック上での再生時間順に並んでいる。

【0044】Track durationはそのエントリが管理する区間のトラック上での再生時間、Media timeはそのエントリが管理する区間の先頭に対応するメディアデータ上の位置、Media rateはそのエントリが管理する区間の再生スピードを表す。尚、Media timeが-1の場合は、そのエントリのTrack duration分、そのトラックでのサンプルの再生を停止する。この区間のことをempty editと呼ぶ。

【0045】図11にEdit listの使用例を示す。ここでは、Edit list atomの内容が図11(a)に示す内容であり、さらにサンプルの構成が図11(b)であったとする。尚、ここではi番目のエントリのTrack durationをD(i)、Media timeをT(i)、Media rateをR(i)とする。このとき、実際のサンプルの再生は、図11(c)に示す順に行われる。このことについて簡単に説明する。

【0046】まず、エントリ#1はTrack durationが13000、Media timeが20000、Media rateが1であるため、そのトラックの先頭から13000の区間はサンプル中の時刻20000から33000の区間を再生する。次に、エントリ#2はTrack durationが5000、Media timeが-1であるため、トラック中の時刻13000から18000の区間、何も再生を行わない。

【0047】最後に、エントリ#3はTrack durationが10000、Media timeが0、Media rateが1であるため、トラック中の時刻18000から28000の区間において、サンプル中の時刻0から10000の区間を再生する。

【0048】図12にUser data atomの構成を示す。このatomには、QuickTimeフォーマットで定義されていない独自の情報を任意個数格納することができる。1個の独自情報は1個のエントリで管理され、1個のエントリはSizeとTypeとUser dataで構成される。Sizeはそのエントリ自体のサイズを表し、Typeは独自情報をそれぞれ区別するための識別情報、User dataは実際のデータを表

す。

【0049】次に、録画中の電源遮断等に対応するために、QuickTimeファイルフォーマットに導入された概念であるFragmented Movieについて説明する。Fragmented movieは、QuickTime フォーマットの1アプリケーションであるMotion JPEG2000で導入された概念であり、上述のSample table atomに相当する情報を、部分的なAVストリーム毎に管理することが可能となっている。

【0050】Fragmented movieを導入したQuickTimeファイルの全体構成を図13に示す。先頭に、そのファイル全体に共通する情報を管理するMovie atomが配置され、その後、Movie fragmentを格納するMovie data atomと、そのMovie fragmentを構成するサンプルのアドレスやサイズ、再生時間等を管理するMovie fragment atomとが交互に配置される。尚、AVストリームデータは、通常のQuickTimeファイルと同様、別ファイルに存在しても構わない。

【0051】録画時にはこの順番で記録を行なっていくことにより、録画時の電源切断による被害を最小限に防ぐことが可能となっている。Movie atomには、そのQuickTimeムービーがFragmented movieであることを示すためのMovie extends atomが含まれる。Movie extends atomには、そのムービーに含まれる各トラックに関するデフォルト値が格納される。

【0052】また、Movie fragment atomには、そのMovie fragment atomが管理するMoviefragmentに関する管理情報が含まれている。管理情報には、その管理するMoviefragment全体に関する情報を格納するMovie fragment header atomと、Movie fragment中の各トラックに関する情報を格納するTrack fragment atomとがある。

【0053】Track fragment atomは、それが管理するトラックに属するMovie fragmentに関する情報を格納するTrack fragment header atomと、そのトラックに属するMovie fragmentを構成する論理的な連続領域(Track runと呼ばれる)をそれぞれ管理するTrack fragment run atomとを含む。以下に、各atomについて詳しく説明する。

【0054】Movie extends atomの構成を図14に示す。Movie extends atomは、前述のように、このatomを含むQuickTimeムービーがFragmented movieであることを示す役割を持つ。

【0055】Track extends atomの構成を図15に示す。Track extends atomは、このQuickTimeムービーに含まれる各トラックのサンプルのデフォルト値を設定するために存在する。track IDはMovie atom中で定義されているトラックのtrack IDを参照する。default-sampleで始まるフィールドは、このatomで管理されるtrack fragmentのデフォルト値を設定する。

【0056】Movie fragment atomの構成を図16に示す。録画中に逐次記録される管理情報はこのatomであ

る。このatomは、前述のとおり、このatomの管理するMovie fragmentに関する実際の情報を格納するatomであるMovie fragment header atomやTrack fragment atomを含む。

【0057】Movie fragment header atomの構成を図17に示す。このatomに格納されている主な情報はsequence-numberである。sequence-numberは、このatomが含まれるMovie fragment atomが管理するMovie fragmentの先頭からの順番を表す。

【0058】Track fragment atomの構成を図18に示す。Track fragment atomは、Moviefragmentに含まれる特定のトラックのサンプルに関する管理情報であるTrack fragment header atomやTrack fragment run atomを格納する。

【0059】Track fragment header atomの構成を図19に示す。このatomは、Movie fragmentに含まれる特定のトラックのサンプルに関するデフォルト値等を格納する。track-IDは、Movie atom中で定義されているトラックのtrack IDとの対応を示す。sample-description-indexは、このatomが管理するサンプルの参照するsample description tableのインデックス番号、default-sampleで始まるフィールドは、それぞれこのatomが管理するサンプルのデフォルト値である。

【0060】Track fragment run atomの構成を図20に示す。このatomは、Track runと呼ばれる、このatomの管理する連続領域や個々のサンプルの管理情報を格納する。sample-countはTrack runに含まれるサンプルの個数を示す。data-offsetはbase-data-offsetからのTrack runのオフセット値を示す。sampleで始まるフィールドはこのatomが管理するサンプルの再生時間等の値を格納する。ただし、上述のデフォルト値と同じであれば、省略してデータサイズを縮小することが可能となっている。

【0061】＜インデックス・ファイル＞ディスク内に含まれるQuickTimeムービーを管理するため、AVインデックス・ファイルという特別のQuickTimeムービーファイルをディスク内に1個置く。

【0062】AVインデックス・ファイルには、ディスク内のファイル(QuickTimeムービーやQuickTimeムービーから参照されている静止画等)に関するサムネイルや各種属性が登録されている。各種属性の中には、そのファイルが外部参照されている回数を示すlink countがある。

【0063】link countを参照することで、そのファイルを参照しているファイルがあるかどうかを容易に知ることができ、他から参照されているファイルを不用意に削除してしまうことを防ぐことができる。

【0064】＜実施例＞本発明の一実施例について、図21乃至図31を用いて説明する。

【0065】＜AVストリームの形態＞まず、本実施例に

におけるAVストリームの構成について、図21及び図22を用いて説明する。AVストリームは整数個のRecord Unit (RU) で構成される。RUはディスク上で連続的に記録する単位である。RUの長さは、AVストリームを構成するRUをどのようにディスク上に配置してもシームレス再生（再生中に画像や音声途切れなく再生できること）やリアルタイムアフレコ（アフレコ対象のビデオをシームレス再生しながらオーディオを記録すること）が保証されるように設定される。この設定方法については後述する。

【0066】また、RU境界がECCブロック境界と一致するようにストリームを構成する。RUのこれらの性質によって、AVストリームをディスクに記録した後も、シームレス再生を保証したまま、ディスク上でRU単位の配置を容易に変更することができる。

【0067】RUは、整数個のVideo Unit (VU) とPost Recording Unit (PRU) で構成される。VUは単独再生可能な単位であり、そのことから再生の際のエントリ・ポイントとなりうる。PRUは同じRUに含まれるVUと同期再生するデータを記録するための領域である。

【0068】VU構成を図22に示す。VUは、1秒程度のビデオデータを格納した整数個のGOP（グループ・オブ・ピクチャ）と、それらと同じ時間に再生されるメインオーディオデータを格納した整数個のAAU（オーディオ・アクセス・ユニット）とから構成される。

【0069】尚、GOPは、MPEG-2ビデオ規格における画像圧縮の単位であり、複数のビデオフレーム（典型的には15フレーム程度）で構成される。AAUはMPEG-1 Layer-II規格における音声圧縮の単位で、1152点の音波形サンプル点により構成される。サンプリング周波数が48kHzの場合、AAUあたりの再生時間は0.024秒となる。VU中では、AV同期再生のために必要となる遅延を小さくするため、AAU、GOPの順に配置する。

【0070】また、VU単位で独立再生を可能とするために、VU中のビデオデータの先頭にはSequence Header (SH) を、末尾にはSequence End Code (SEC) を置く。VUの再生時間は、VUに含まれるビデオフレーム数にビデオフレーム周期をかけたものと定義する。さらに、VUを整数個組み合わせる場合、RUの始末端をECCブロック境界に合わせるため、VUの末尾を0で埋める。

【0071】PRUの領域サイズは、ここではRUの再生時間にオーディオの最大ビットレートをかけたものとする。

【0072】＜AVストリーム管理方法＞上記のAVストリームの管理情報の構成について説明する。AVストリームは、図23に示すように、ムービーファイル2401と、ムービーファイル2401の管理情報のバックアップに相当する管理情報バックアップファイル2402とで管理される。

【0073】ムービーファイル2401は、Movie data ato

に格納されたAVストリームと、AVストリームを構成するサンプルのアドレスやサイズ、再生時間等を管理するMovie atomとで構成される。AVストリームは、前述のように、RUで構成され、各RUは必ずディスク上で連続的に配置されるように記録される。

【0074】一方、管理情報バックアップファイル2402は、ムービーファイル2401中の各RUを管理するMovie fragment atomで構成される。この2つのファイルを、図23に示すように、ディスク上においてRU単位で多重化する。このような構成を取る理由を以下に説明する。

【0075】図13のような構成を取った場合、光ディスクにおいては録画時には都合が良いが再生時には都合が悪い。なぜならば、録画時にはシークを行うことなく、その時点までの管理情報をMovie fragment atomに記録できるのに対し、再生時には、Movie data atom#1を再生するためには、まずMovie fragment atom#1を読み込む必要があるため、ピックアップを行ったり来たりさせなければならず、読み込みの効率が悪くなるからである。

【0076】このとき、図23に示すような構成を取ることで、録画中の電源断やムービーファイル2401のMovie atomの破損時のバックアップが可能となると同時に、Movie fragment atomを用いない場合と同様、再生時の効率的な管理情報の読み込みが可能となる。

【0077】尚、上記の2ファイルはディスク上で多重化して記録するが、全体をディスク上で連続的に記録する必要はなく、各Record Unitと直前のMovie fragment とが連続的に記録されていれば良い。つまり、図24に示す不連続可能点以外ではディスク上で連続的に記録する。このような連続記録を行う理由は、アフレコ時においてPRUへの記録の直前に、アフレコに伴って必要となるMovie fragmentの更新を、シークが加わることなく可能にするためである。

【0078】また、図23においては、Movie fragment atomとRUとが接しているように記載されているが、実際にはMovie fragment atomとRUとの間には、Movie fragment atomの先頭からRUの直前までのバイト数が1ECCブロックサイズの整数倍になるように、パディングとしてFree space atomを挿入する。Free space atomとは、QuickTimeファイルフォーマットで規定されているatomで、領域の確保に利用される。

【0079】次に、ムービーファイル2401の管理情報の構成について説明する。ムービーファイル2401の管理情報は、ビデオデータ用のビデオトラック、メインオーディオ用のメインオーディオトラック、アフレコ領域管理用のアフレコ領域トラックおよび実際にアフレコされたデータを管理するためのアフレコオーディオトラックで構成される。

【0080】ビデオトラックは、Sequence headerからS



equence end codeまでのビデオデータを1サンプル、VU中のビデオの塊を1チャンクとして管理する。メインオーディオトラックは、AAUを1サンプル、VU中のオーディオの塊を1チャンクとして管理する。アフレコ領域トラックは、PRUに所定のビットレートでオーディオデータを記録したものとして、AAUを1サンプル、PRUを1チャンクとして管理する。

【0081】ただし、このトラックはどのタイミングのオーディオデータをどこに記録するかを知らせるためのものであり、誤って再生しないよう、Track header atomのflagを再生不可に設定する。アフレコオーディオトラックは、PRUに実際に記録されているオーディオデータに関して1AAUを1サンプル、PRU中の連続記録データを1チャンクとして管理する。

【0082】次に、管理情報バックアップファイル2402の管理情報の構成について説明する。ムービーファイル2401と同様、ビデオトラック、メインオーディオトラック、アフレコ領域トラック、アフレコオーディオトラックで構成し、AAUを1サンプル、Sequence headerからSequence end codeまでのビデオデータを1サンプルとして

管理する。

【0083】バックアップ管理情報ファイルにおけるPRUの管理方法について、図25に示す構成のRUを例にとって説明する。ここでは、RUはムービーファイルの先頭からLバイトの位置にあるとする。また、PRUは、10個のAAUで構成され、先頭から4個のAAUは未アフレコ、後半の6個のAAUはアフレコ済みとする。そしてまた、各AAUは再生時間0.024秒、サイズは768バイトであるとする。

【0084】このとき、図26に示すように、Movie extends atom中のアフレコ領域トラックおよびアフレコオーディオトラックのdefault-sample-sizeは768、default-sample-durationは2160（1秒の細かさであるTimescaleを90000とした場合）となる。

【0085】また、このMovie fragment atom中のアフレコ領域トラックおよびアフレコオーディオトラックの内容は、図27に示すとおりになる。まず、アフレコ領域トラックに関しては、PRU中の全サンプルを管理するため、sample-countは10となる。

【0086】次に、アフレコオーディオトラックに関しては、アフレコ済みのサンプルのみを管理するため、sample-countは6となり、data-offsetはPRUの先頭からアフレコ済みの先頭サンプルまでのバイト数となる。アフレコを行う際には、アフレコオーディオトラックのこれらの値を書き換えることになる。

【0087】<Record Unit再生時間決定方法>まず、アフレコ対応ストリームにおけるRU再生時間の決定方法について説明する。この決定方法では、機器間での互換性確保のため、基準となるデバイス（リファレンス・デ

バイス・モデル）と基準となるアフレコアルゴリズム（リファレンス・アフレコ・アルゴリズム）とを想定し、次にそれらを用いてアフレコを行った際にシームレス再生が破綻しないように、RU再生時間を決める。

【0088】それではまず、リファレンス・デバイス・モデルについて、図28を用いて説明する。リファレンス・デバイス・モデルは、1個のピックアップとそれにつながるECCエンコーダ・デコーダ501、トラックバッファ502、デマルチプレクサ503、アフレコ用バッファ504、オーディオエンコーダ509、ビデオバッファ505、オーディオバッファ506、ビデオデコーダ507、オーディオデコーダ508より構成される。

【0089】本モデルでは、ピックアップが1個であるため再生用データのディスクからの読み出しとアフレコ用データのディスクへの記録は時分割で行う。ディスクから再生用データを読み出す際、PRUと直前のMovie fragment atomを含めて読み出す。読み出されたMovie fragment atomとPRUを含むECCブロック（PRUブロック）は、トラックバッファ502からアフレコ用バッファ504に送られる。

【0090】オーディオエンコーダ509は、AAU周期でアフレコ用バッファ504に出力する。この出力によって、アフレコ用バッファ504中の対応するPRUブロックを上書きする。アフレコデータの記録は、PRUブロックを所定のECCブロックに記録することで行う。

【0091】本モデルにおいて、Movie fragment atomとPRUブロックをトラックバッファ502からアフレコ用バッファ504に送ることを想定しているのは、Movie fragment atomおよびPRUの書き換えに伴うECCブロックの再度読み出しを省略するためである。

【0092】本モデルにおけるシームレス再生は、VUのデコード開始時にトラックバッファ502上に少なくとも1個VUが存在すれば保証されるものとする。ECCデコーダ501からデータの出力速度はRsとする。また、アクセスによる読み出し、記録の停止する最大期間をTaとする。

【0093】尚、これら期間にはシーク時間、回転待ち時間、アクセス後に最初にディスクから読み出したデータがECCから出力されるまでの時間が含まれる。本実施例ではRs=20Mbps、Ta=1秒とする。

【0094】本実施例におけるリファレンス・アフレコ・アルゴリズムについて、図29を用いて説明する。尚、図29中の（1）から（8）までの番号は、以下の説明中の（1）から（8）までの番号に対応する。アルゴリズムの概要は次の通りである。

【0095】（1）再生用データの読み出しを行う。  
（2）N番目のPRUであるPRU#Nに対応するオーディオデータのエンコードが終了すると同時にMovie fragment atom#N-1へのアクセスを行う。（3）Movie fragment atom#N-1とPRU#Nに対応するPRUブロックをディスクに記録する。（4）元の読み出し位置に戻る。（5）再生用デ



ータの読み出しを行う。読み出しの際にRU境界を跨ぎ、分断のジャンプが発生するが、そのまま読み出しを続ける。

【0096】(6) N+1番目のPRUであるPRU#N+1に対応するオーディオデータのエンコードが終了すると同時に、Movie fragment atom#Nへのアクセスを行う。

(7) Movie fragment atom#NとPRU#N+1に対応するPRUブロックをディスクに記録する。(8) 元の読み出し位置に戻る。以上の動作を繰り返す。

【0097】前記リファレンス・デバイス・モデルにおいて、前記リファレンス・アフレコ・アルゴリズムを用いてアフレコを行った場合、次のような条件を満たせ \*

$$\sum_{i=1}^n Te(i) \geq \sum_{i=1}^n (Tr(i) + Tw(i)) \cdots \text{<式 2>}$$

【0101】を満たす十分条件であるためである。

【0102】また、PRUエンコード完了に同期してアフレコデータのディスクへの記録を行っているため、アフレコ用バッファ504中のデータが累積していくことはなく、アフレコ用バッファ504のオーバーフローもない。 ※20

$$Tr(i) = Te(i) \times (Rv + Ra + Rp) / (Rs + Ta + Ly / Rs) \cdots \text{<式 3>}$$

となる。

【0104】右辺第1項はRU#iの読み出し時間を表す。右辺第2項はRU#i読み出し直後に発生する分断ジャンプによる最大アクセス時間を表す。右辺第3項は、Movie fragment atom読み出し時間を表す。

【0105】尚、Movie fragment atomのサイズはRU再生時間と比例関係にあるが、録画時の電源断に対応した10秒程度ではそのサイズが1 ECCブロックを越えることはないため、Movie fragment atomの読み出し時間は1 ECCブロック読み出し時間とした。

【0106】また、Tw(i)は、  
 $Tw(i) = 2Ta + Te(i) \times Rp / (Rs + Ly / Rs + Ly / Rs) \cdots \text{<式 4>}$   
 である。

【0107】ここで、右辺第1項はRUへの往復アクセス時間を示す。PRUへの往復のアクセス時間に最大アクセス時間Taを用いているのは、以下の理由に基づく。

【0108】現在読み出ししているトラックと記録すべきPRUの存在するトラックの距離は、そのときの再生用バ ★

$$Te(i) \geq (3Ta \times Rs + 3Ly) / (Rs - Rv - Ra - 2Rp) \cdots \text{<式 5>}$$

が得られる。

【0111】つまり、アフレコ保証可能なRU再生時間下限値Teminは、

$$Temin = (3Ta \times Rs + 3Ly) / (Rs - Rv - Ra - 2Rp) \cdots \text{<式 6>}$$

となる。  
 【0112】このとき、RU再生時間の上限値Tmaxを次のように設定する。

$$Tmax = Temin + Tvmax \cdots \text{<式 7>}$$

ここで、TvmaxはVUの最大再生時間である。上限値を設定するのは、次の理由に基づく。Teが大きくなるにした

※ば、トラックバッファ502のアンダーフローがないことを保証することができる。

【0098】その条件とは、AVストリーム中の任意のRUであるRU#iについて最大再生時間をT(i)、分断ジャンプを含めた最大読み出し時間をTr(i)、RU#i中のPRUの最大記録時間をTw(i)としたとき、

$$Te(i) \geq Tr(i) + Tw(i) \cdots \text{<式 1>}$$

が成立することである。

【0099】なぜなら、この式は、シームレス再生の十分条件である任意のnにおける

【0100】

【数1】

※【0103】<式1>中のTr(i)は、AVストリーム中のメインオーディオとサブオーディオ、ビデオの最大ビットレートをそれぞれRa、Rp、Rv、ECCブロックサイズをLyとしたとき、

★ ッファによる遅延時間に依存する。しかし、遅延時間は再生用バッファサイズによって異なり、また同じバッファサイズであっても、直前に衝撃によって読み出しが一時的に停止した場合にも異なる。

【0109】すなわち、アクセスする距離は不定であり、そのため最悪値で見積もる必要がある。右辺第2項は、RU#iに含まれるPRUをディスクに記録するための時間の合計を表す。右辺第3項はPRU両端が含まれるECCブロック中のアフレコデータ以外の記録時間の最大値を表している。このような項が必要な理由は、PRUの両端はECCブロック境界と一致しているとは限らないため、PRU記録時には、PRUのサイズより最大1 ECCブロック分多く記録することになるためである。右辺第4項はMovie fragment atomの記録に要する時間を表す。

【0110】このとき、<式1>に<式3>と<式4>を代入して、Te(i)で解くと、リアルタイムアフレコを保証可能なTe(i)の条件

が得られて、アフレコ時において図29の(2)から(8)までの期間が長くなる。この間は再生用データのディスクからの読み出しができないため、再生を継続するためには、Teの増加に応じて、トラックバッファ502のサイズを増やす必要がある。

【0113】上限値を設定するのは、このとき必要となるトラックバッファ502のサイズを見積り可能にするためである。また、下限値と上限値の間にVUの最大再生時間分のマージンがあることにより、任意の再生時間の組み合わせでRUを構成することが可能である。

【0114】尚、ここでは最大再生時間をAVストリームのビットレートに応じて設定しているが、可能な最大のビットレートに基づき、AVストリームのビットレートに関わらず一定としても良い。

【0115】＜記録時の処理＞ユーザから録画が指示された場合の処理を、図30とともに説明する。このとき記録するAVストリームは、ビデオのビットレート $R_v=5\text{Mbps}$ 、メインオーディオのビットレート $R_a=256\text{kbps}$ 、アフレコオーディオのビットレート $R_p=256\text{kbps}$ で、VU再生時間 $T_v\approx 0.5$ 秒固定アフレコ対応ストリームとする。また、ファイルシステムの管理情報はすでにRAM上に読み込まれているものとする。

【0116】まず、ストリームの構成や連続領域の構成を決定する（ステップ701）。1VUを1GOP15フレームで構成するとしたとき、＜式6＞＜式7＞に $R_s=20\text{Mbps}$ 、 $T_a=1$ 秒、 $R_v=5\text{Mbps}$ 、 $R_a=256\text{kbps}$ 、 $R_p=256\text{kbps}$ 、 $T_{vmax}\approx 0.5$ 秒を代入し、 $T_e(i)$ の範囲4.4秒以上4.9秒以下が得られる。

【0117】 $T_{vmax}\approx 0.5$ 秒でこの条件を満たすのは、 $T_e(i)=4.5$ 秒のときとなり、9個のVU毎にPRUが挿入されることになる。MPEG-1 audio layer-IIにおいて、ビットレート256kbpsのとき、AAUの再生時間 $T_{af}$ は0.024秒、サイズは768byteとなり、このときのPRUの領域サイズは、144384byteとなる。また、連続領域には1個のPRUと9個のVUと1個のMovie fragment atomとが含まれるようにする。

【0118】まず、9個のVUと1個のPRUを連続的に記録可能な空き領域を探す。Movie fragment atomを考慮しない理由は、先頭のRUだけはMovie fragment atomと連続的に記録する必要がないためである。具体的には、 $9 \times T_{vmax} \times (R_a + R_v) + 9 \times T_{av} \times R_a$ 、つまり24.8Mbit以上の連続的な空き領域をRAM102上のSpace Bitmapを参照して探す。存在しなければ、録画を中止し、録画できないことをユーザに知らせる（ステップ702）。

【0119】また、オーディオエンコーダ117、ビデオエンコーダ118をそれぞれ起動する（ステップ703）。そして、記録用バッファに1ECCブロック分(32KB)以上のデータが蓄積されているかどうかをチェックし（ステップ704）、蓄積されている間、ステップ705からステップ708を繰り返す。

【0120】蓄積されていれば、次に記録するディスク上のECCブロックの空き状況をRAM上のSpace Bitmapを参照して調べる（ステップ705）。空きがなれば、9個のVUとPRUと1個のMovie fragment atomを記録可能な連続的な空き領域を探し（ステップ707）、その空き領域の先頭へピックアップを移動する（ステップ708）。

【0121】そして、記録用バッファ111中の1ECCブロック分のデータをディスクに記録する（ステップ706）。記録用バッファ111にデータが蓄積されていないければ、記録終了が指示されているかどうかをチェックし

（ステップ709）、記録終了でなければ、ステップ704を実行する。

【0122】記録終了が指示されていた場合、以下のステップを実行する。まず、記録用バッファ中の32KBに満たないデータに関して、末尾にダミーデータを付加し32KBにする（ステップ710）。次に、そのデータをディスク上に記録する（ステップ711～ステップ714）。RAM102上のQuickTime管理情報とファイルシステム管理情報とを光ディスク106に記録する（ステップ715～ステップ716）。

【0123】以上の処理と並行するオーディオエンコーダ117、ビデオエンコーダ118やマルチプレクサ113の動作について説明する。それぞれのエンコーダはマルチプレクサ113にエンコード結果を送り、マルチプレクサはそれらを多重化用バッファ114に格納する。

【0124】1VU分のデータ、つまり1GOPとそれに同期して再生されるAAUが多重化用バッファ114に蓄積されたら、マルチプレクサ113は記録用バッファ111に1VUのデータを送る。このとき、そのVUが9×i番目（iは0以上の整数）のVUであったら、上述のサイズを持ったPRUを先に記録用バッファ111に送る。

【0125】さらに、ホストCPU101に1VU分のデータがエンコードできたことを通知し、ホストCPU101はVUを構成するGOPやAAUの数およびサイズを基にRAM102上のQuickTime管理情報を更新する。

＜アフレコ時の処理＞ユーザからアフレコが指示された場合の処理を図31とともに説明する。ここで、アフレコの対象となるAVストリームに関するQuickTime管理情報は、すでにRAM102に読み込まれているものとする。

【0126】まず、アフレコ開始位置を含むディスク上のVUの先頭から再生用データの読み出しを行う（ステップ802）。このとき、十分な再生時間分のデータを読み出すまでステップ802を繰り返す（ステップ803）。ここで十分とは、再生用データ読み出しの中断期間が最大の場合でも再生が途切れないだけのデータ量を意味する。

【0127】また、PRUを読み出した際には、Movie fragment atomおよびPRUを含むECCブロックをアフレコ用バッファ111に送る。このとき、アフレコ用バッファ111中のPRUを管理するために、アフレコ用バッファ111中の各PRUの再生開始時間（AVストリームの先頭からの相対時間）とアフレコ用バッファ111中でのアドレスの組をテーブルとしてRAM102に保持する。

【0128】次に、ビデオデコーダ116とオーディオデコーダ115および、オーディオエンコーダ117を起動する（ステップ804）。オーディオエンコーダ117は、サンプリングされた音声波形をAAUにエンコードし、AAUの周期でマルチプレクサ113に送る。その際に、各AAUについてAVストリームの先頭からの相対時間を付加する。

【0129】マルチプレクサ113は、AAUに付加された時間に基づき、AAUをアフレコ用バッファ111中のPRUに格

納する。また、ステップ802で読み出したMovie fragment atom中のアフレコオーディオトラックの内容を更新し、アフレコ用バッファ111に格納する。RU中の最後のPRUにCAUを最後まで格納し終わったら、ホストCPU101にCRUのエンコード終了を通知する。

【0130】次に、ユーザからアフレコ終了を指示されていないかチェックする(ステップ805)。指示されていないければ、PRUのエンコードが終了するまで、ステップ802と同様に、再生用データの読み出しを行う(ステップ809)。

【0131】マルチプレクサ部からRUエンコード終了が通知されたら(ステップ806)、RAM102上のテーブルに保持しているそのRUに含まれるPRUの再生開始時間から、QuickTime管理情報を用いて、そのPRUを記録すべき光ディスク106上のアドレス、つまり元々そのPRUが記録されていたアドレスを求める。

【0132】アフレコを開始したRUのPRU記録の時以外は、求めたPRUのアドレスから1 ECC分の引くことで直前のMovie fragment atomのアドレスを求める。そのアドレスにピックアップ107を移動させ(ステップ807)、Movie fragment atomとPRUを含むECCブロックを光ディスク107に記録する(ステップ808)。

【0133】アフレコ終了が指示されていれば、現在エンコード中のPRUのエンコード完了を待って(ステップ810)、そのPRUの直前のMovie fragment atomの記録アドレスを求めピックアップを移動し(ステップ811)、Movie fragment atomとPRUを記録する(ステップ812)。最後にQuickTime管理情報をディスクに記録する(ステップ813)。

【0134】尚、本実施例においては、PRU中のアフレコ済みデータの管理を、Movie fragment atomで行っているが、バックアップ管理情報ファイル2402のMovie atom#2で行うことも考えられる。すなわち、バックアップ管理情報ファイル2402において、アフレコオーディオトラックのみ通常のSample table atomで管理することになる。

【0135】この構成においても、ムービーファイル2401のアフレコ済みデータの管理情報と同様の内容は、バックアップ管理情報ファイル2402に存在するため、ムービーファイル2401が破損した場合も、バックアップ管理情報ファイル2402を参照することで、どの領域がアフレコ済みかを知ることが可能である。

【0136】また、アフレコ済みデータの管理情報は、バックアップ管理情報ファイル2402のMovie atomにまとまって存在するため、管理情報の更新は短時間で可能である。ただしこの構成では、PRUとPRU中のアフレコ済みデータの管理情報とがディスク上で離れた位置に存在することになるため、アフレコ時に管理情報を更新することは困難となり、本実施例のようにアフレコ時の電源切断に対応することはできない。

【0137】しかし、本実施例に比べて、次のようなメリットがある。すなわち、このような構成を取ることによって、アフレコ時のMovie fragment atomの挿入間隔とPRUの挿入間隔を独立に設定することが可能となる。例えば、Movie fragment atomをPRUの挿入間隔より短い間隔で挿入することで、初回ビデオ記録時の電源切断に対するユーザデータ保護を強力にすることが可能となる。

【0138】また、フラグメンテーションを解消するため、ディスク上でデータの再配置を行うことを考慮した場合、本実施例ではムービーファイル2401中のRecord Unitとバックアップ管理情報ファイル2402のMovie fragment atomとがディスク上で連続であることを保つ必要があるが、このような構成の場合、ムービーファイル2401がRecord Unit単位に連続的に記録されることだけを気にすればよく、バックアップ管理情報ファイル2402に関しては気にする必要がない。

【0139】

【発明の効果】以上説明したように、本発明によれば、アフレコ用領域とバックアップ管理情報をディスク上で連続的に配置することにより、アフレコ時においてバックアップ管理情報を更新することが容易となる。

【図面の簡単な説明】

【図1】本発明の実施形態における概略構成を示すブロック図である。

【図2】QuickTimeファイルフォーマットにおける管理情報とAVストリームとの関係を示す説明図である。

【図3】QuickTimeファイルフォーマットにおけるMovie atomの概要を示す説明図である。

【図4】QuickTimeファイルフォーマットにおけるTrack atomの概要を示す説明図である。

【図5】QuickTimeファイルフォーマットにおけるTrack header atomの構成を示す説明図である。

【図6】QuickTimeファイルフォーマットにおけるMedia atomの構成を示す説明図である。

【図7】QuickTimeファイルフォーマットにおけるMedia information atomの構成を示す説明図である。

【図8】Sample table atomによるデータ管理の例を示す説明図である。

【図9】QuickTimeファイルフォーマットにおけるSample table atomの構成を示す説明図である。

【図10】QuickTimeファイルフォーマットにおけるEdit atomの構成を示す説明図である。

【図11】Edit atomによる再生範囲指定の例を示す説明図である。

【図12】QuickTimeファイルフォーマットにおけるUser data atomの構成を示す説明図である。

【図13】QuickTimeファイルフォーマットにおけるFragmented movieの全体構成を示す説明図である。

【図14】QuickTimeファイルフォーマットにおけるMovie extends atomの構成を示す説明図である。

【図15】QuickTimeファイルフォーマットにおけるTrack extends atomの構成を示す説明図である。

【図16】QuickTimeファイルフォーマットにおけるMovie fragment atomの構成を示す説明図である。

【図17】QuickTimeファイルフォーマットにおけるMovie fragment header atomの構成を示す説明図である。

【図18】QuickTimeファイルフォーマットにおけるTrack fragment atomの構成を示す説明図である。

【図19】QuickTimeファイルフォーマットにおけるTrack fragment header atomの構成を示す説明図である。 10

【図20】QuickTimeファイルフォーマットにおけるTrack fragment run atomの構成を示す説明図である。

【図21】本発明の一実施例におけるアフレコ対応ストリームの構成を示す説明図である。

【図22】本発明の一実施例におけるVUの構造を示す説明図である。

【図23】本発明の一実施例におけるムービーファイルとバックアップ管理情報の関係を示す説明図である。

【図24】本発明の一実施例における連続記録制限を示す説明図である。

【図25】本発明の一実施例におけるバックアップ管理情報によるアフレコ領域の管理方法を示す説明図である。

【図26】本発明の一実施例におけるバックアップ管理情報のMovie dataの例を示す説明図である。

【図27】本発明の一実施例におけるバックアップ管理情報のMovie fragmentの例を示す説明図である。

【図28】本発明の一実施例におけるリファレンス・デバイス・モデルを示す説明図である。

【図29】本発明の一実施例におけるリファレンス・ア \*30

\*フレコ・アルゴリズムを示す説明図である。

【図30】本発明の一実施例における記録動作を示すフローチャートである。

【図31】本発明の一実施例におけるアフレコ動作を示すフローチャートである。

【図32】従来技術における管理情報とAVストリームとの配置を示す説明図である。

【図33】従来技術におけるアフレコ対応AVストリームを示す説明図である。

【符号の説明】

- 100 バス
- 101 ホストCPU
- 102 RAM
- 103 ROM
- 104 ユーザインタフェース
- 105 システムクロック
- 106 光ディスク
- 107 ビックアップ
- 108 ECCデコーダ
- 20 109 ECCエンコーダ
- 110 再生用バッファ
- 111 記録/アフレコ用バッファ
- 112 デマルチプレクサ
- 113 マルチプレクサ
- 114 多重化用バッファ
- 115 オーディオデコーダ
- 116 ビデオデコーダ
- 117 オーディオエンコーダ
- 118 ビデオエンコーダ

【図3】

```
Movie atom {
    Atom size
    Type(='moov')
    Movie header atom
    Track atom (video track)
    Track atom (main audio track)
    :
    User data atom
}
```

【図4】

```
Track atom {
    Atom size
    Type(='trak')
    Track header atom
    Edit atom
    Track reference atom
    Media atom
    User data atom
    :
```

【図5】

```
Track header atom {
    Atom size
    Type(='tkhd')
    Version
    Flags
    Creation time
    Modification time
    Track ID
    Reserved
    Duration
    Reserved
    Layer
    Alternate group
    Volume
    Reserved
    Matrix structure
    Track width
    Track height
}
```

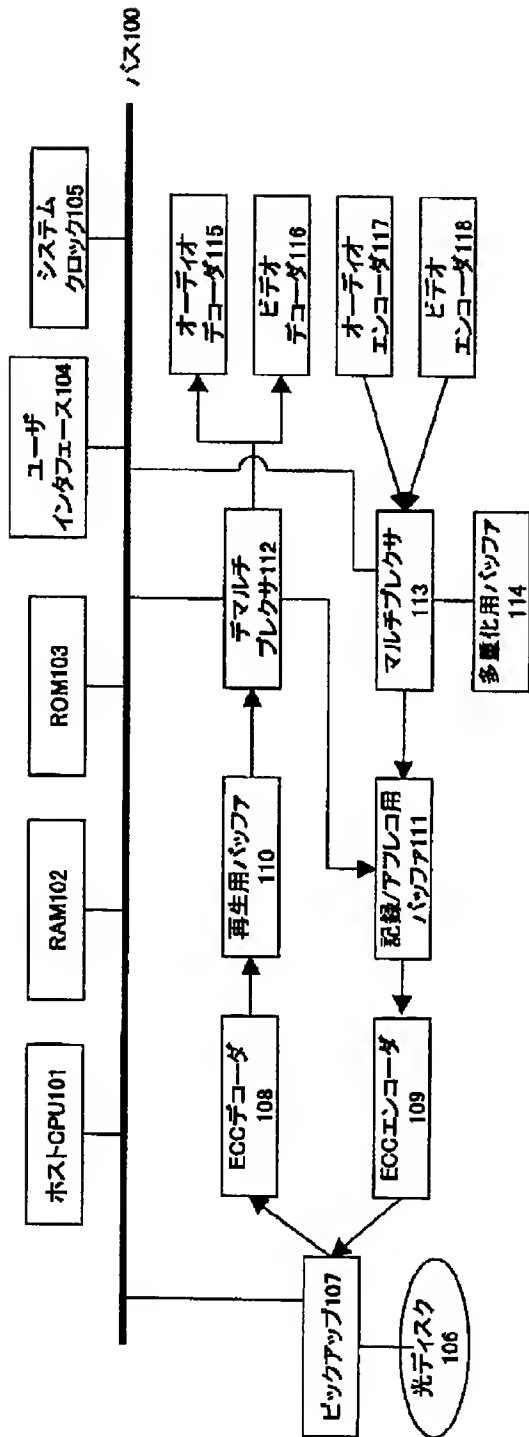
【図6】

```
Media atom {
    Atom size
    Type(='mdia')
    Media header atom
    Handler reference atom
    Media information atom
    User data atom
    :
}
```

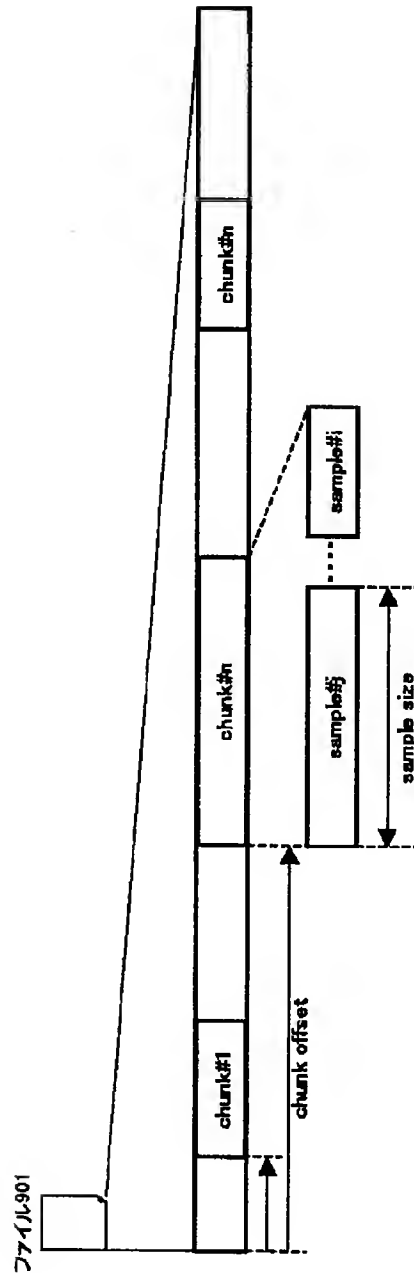
【図17】

```
Movie fragment header atom {
    Atom size
    Type(='mfhd')
    version
    flags
    sequence-number
}
```

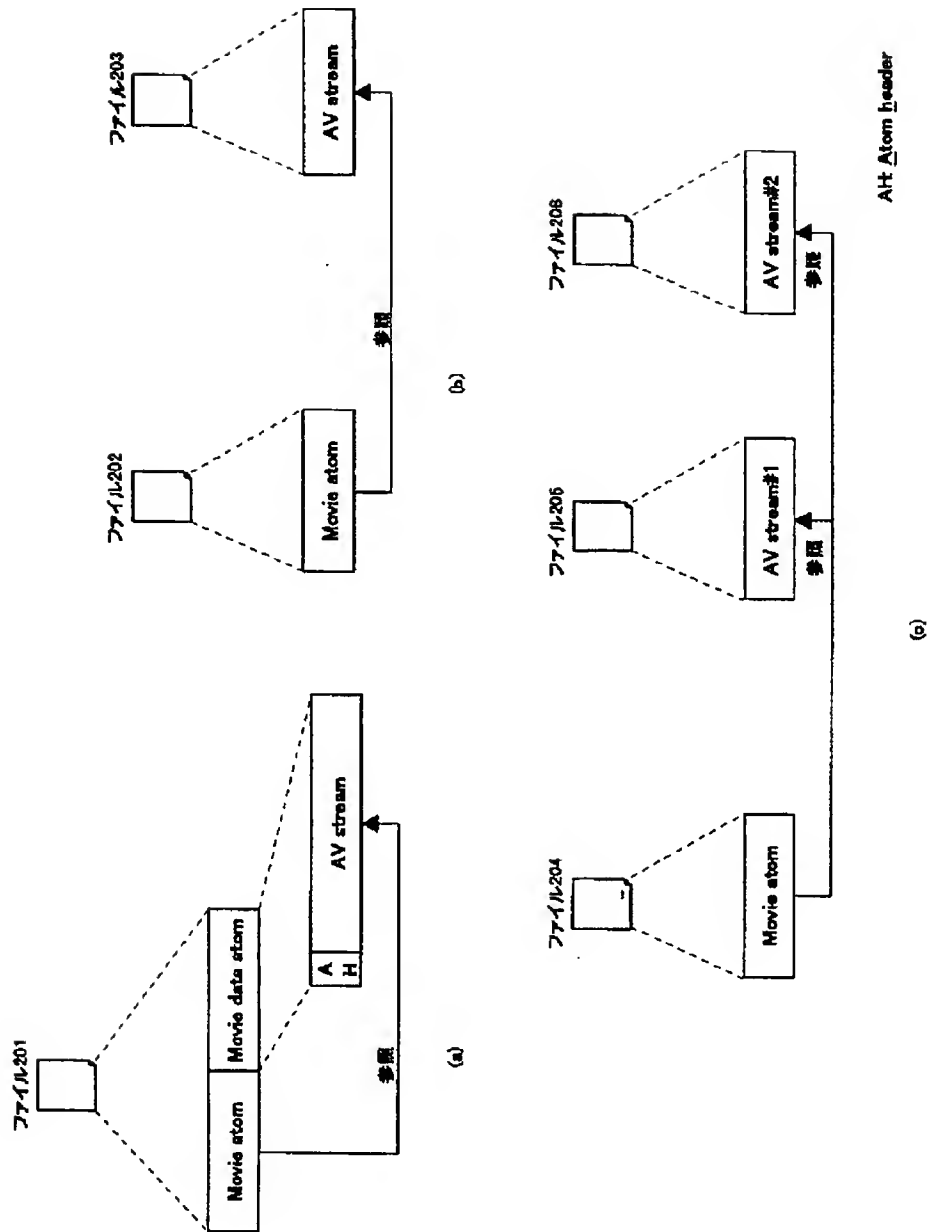
【図1】



【図8】



【図2】



【図7】

```

Media information atom {
    Atom size
    Type(='minf')
    {Video or Sound or Base} media information header atom
    Handler reference atom
    Data information atom
    Sample table atom
}

```

【図9】

```

Sample table atom {
    Atom size
    Type(='stbl')
    Sample description atom
    Time-to-sample atom
    Sync sample atom
    Sample-to-chunk atom
    Sample size atom
    Chunk offset atom
}

```

【図10】

```

Edit atom {
    Atom size
    Type(='edts')
    Edit list atom
}

```

【図12】

```

User data atom {
    Atom size
    Type(='udta')
    for (i=0; i<N; i++){
        Atom size
        Type
        User data
    }
}

```

【図14】

```

Movie extends atom {
    Atom size
    Type(='mvex')
    version
    flags
    for (i=0; i<n; i++){
        Track extends atom
    }
}

```

```

Edit list atom {
    Atom size
    Type(='elst')
    Versions
    Flags
    Number of entries(=N)
    for (i = 0; i < N; i++){
        Track duration
        Media time
        Media rate
    }
}

```

【図16】

```

Movie fragment atom {
    Atom size
    Type(='moof')
    version
    flags
    Movie fragment header atom
    for (i=0; i<n; i++){
        Track fragment atom
    }
}

```

【図15】

```

Track extends atom {
    Atom size
    Type(='trex')
    version
    flags
    track-ID
    default-sample-description-index
    default-sample-duration
    default-sample-size
    default-sample_flags
}

```

【図19】

```

Track fragment header atom {
    Atom size
    Type(='tfhd')
    version
    flags
    track-ID
    base-data-offset
    sample-description-index
    default-sample-duration
    default-sample-size
    default-sample-flags
}

```

【図18】

```

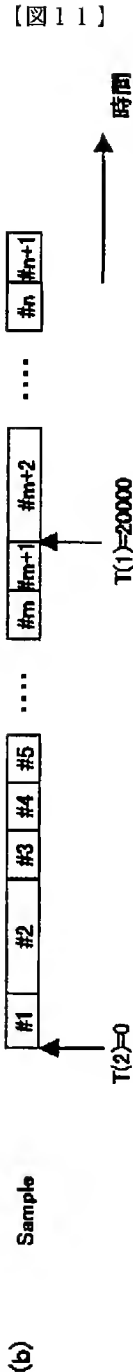
Track fragment atom {
    Atom size
    Type(='traf')
    version
    flags
    Track fragment header atom
    for (i=0; i<n; i++){
        Track fragment run atom
    }
}

```

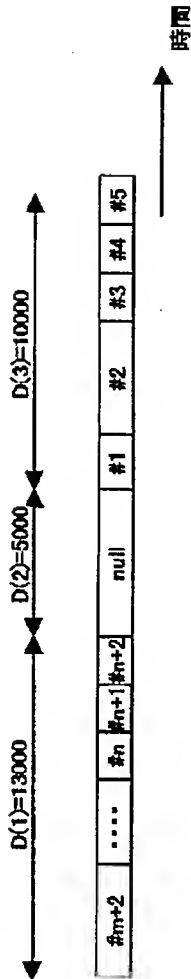


Entry Number	Track duration D(i)	Media time T(i)	Media rate R(i)
#1	13000	20000	1
#2	5000	-1	1
#3	10000	0	1

(a)

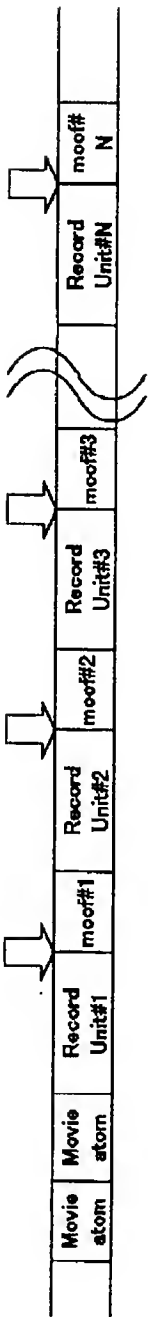


(b)



(c)

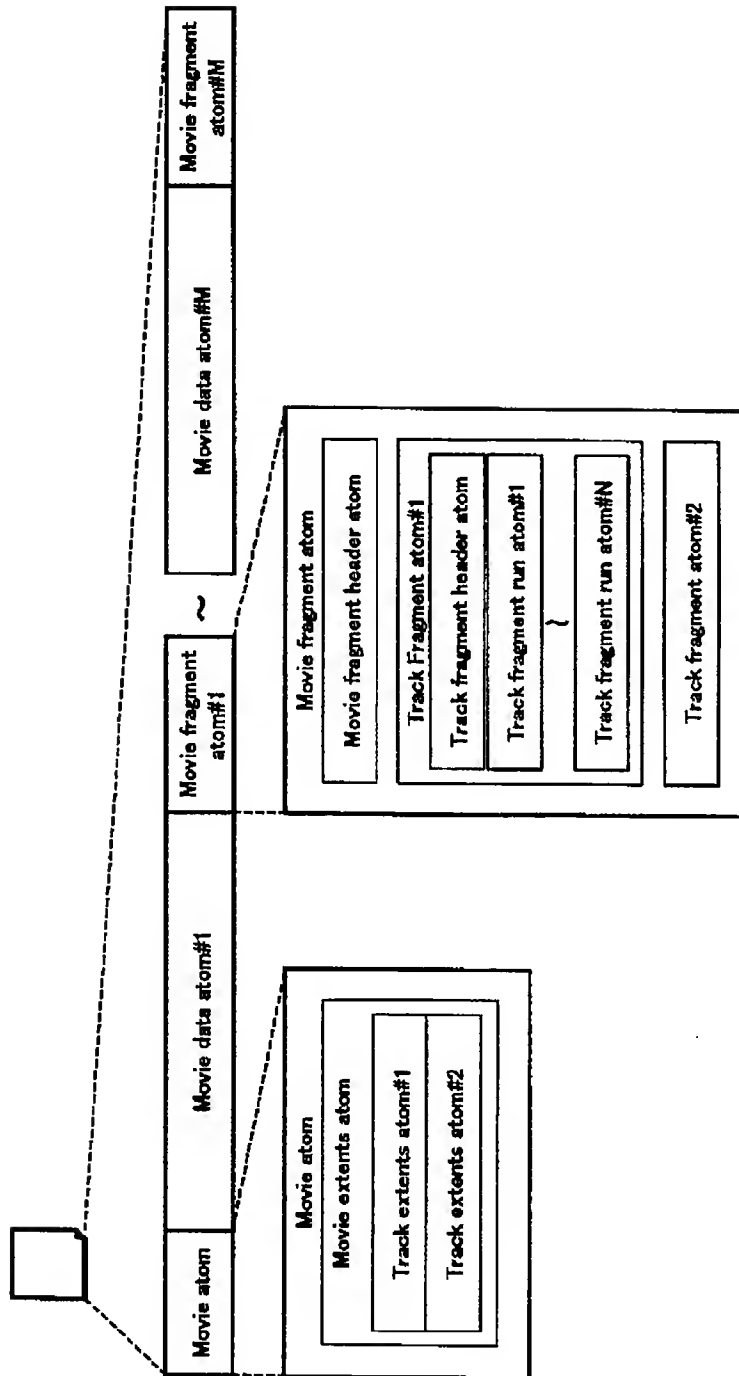
【図24】



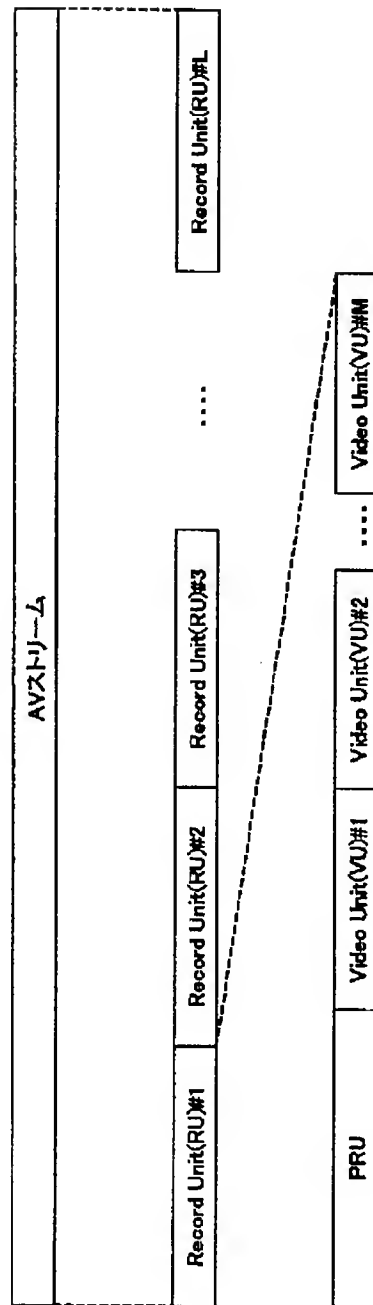
不連続可能点

moof: Movie fragment atom

【図13】



【図21】



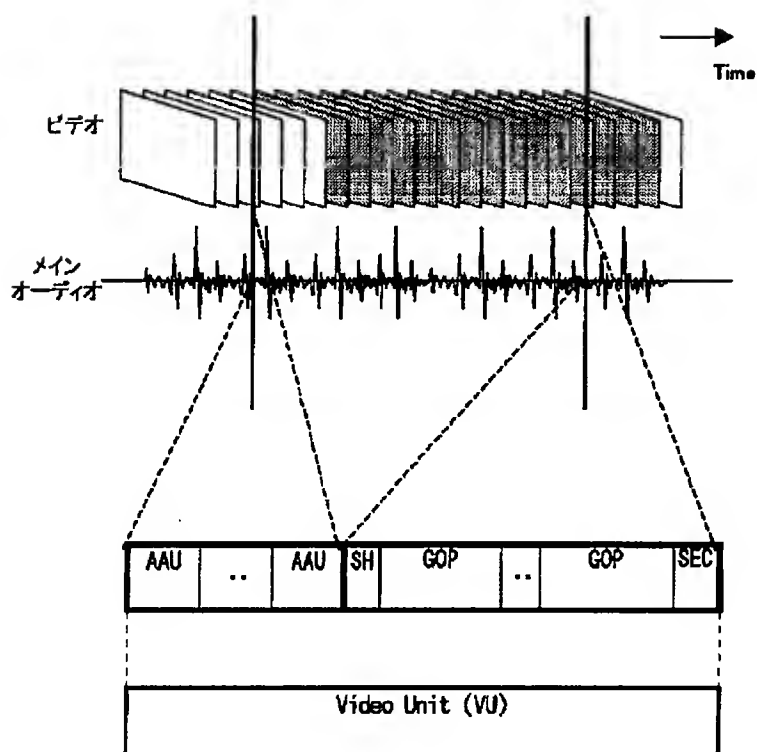
【図20】

```

Track fragment run atom {
  Atom size
  Type(='trun')
  version
  flags
  sample-count(=N)
  data-offset
  first-sample-flags
  for (i=0; i<N; i++){
    sample-duration
    sample-size
    sample-flags
    sample-composition-time-offset
  }
}

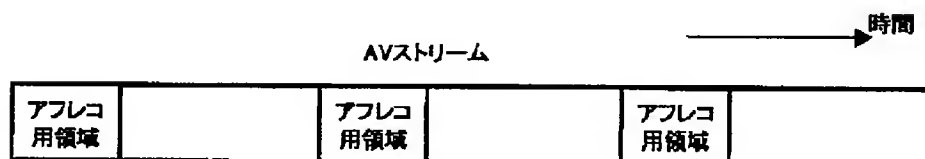
```

【図22】

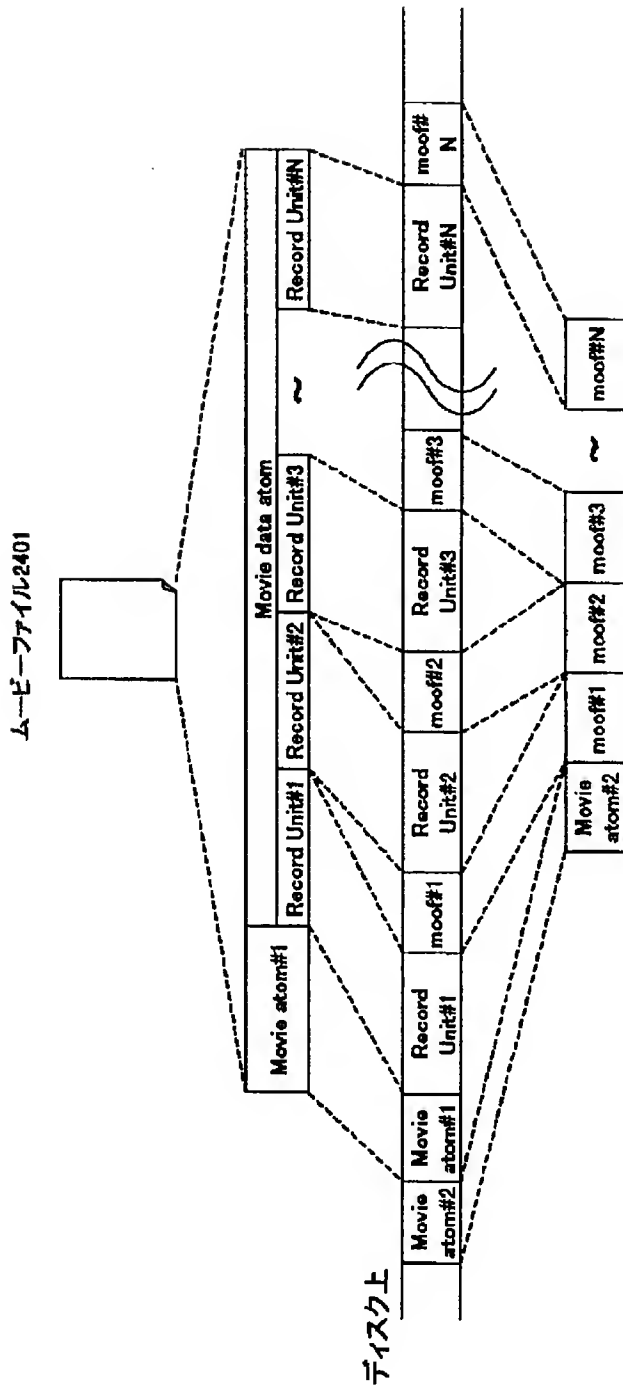


AAU: Audio Access Unit  
 SH: Sequence Header  
 SEC: Sequence End Code

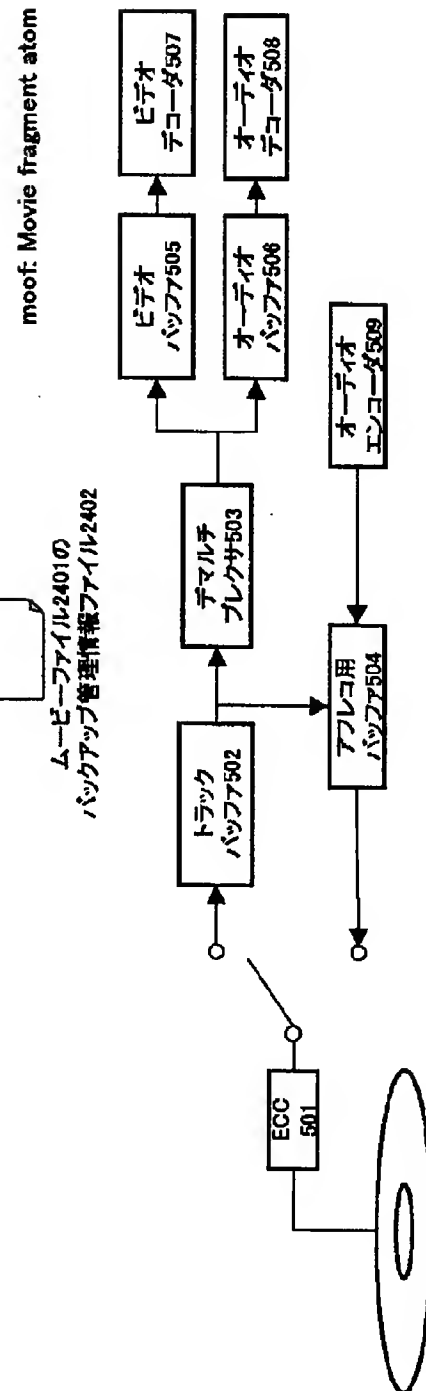
【図33】



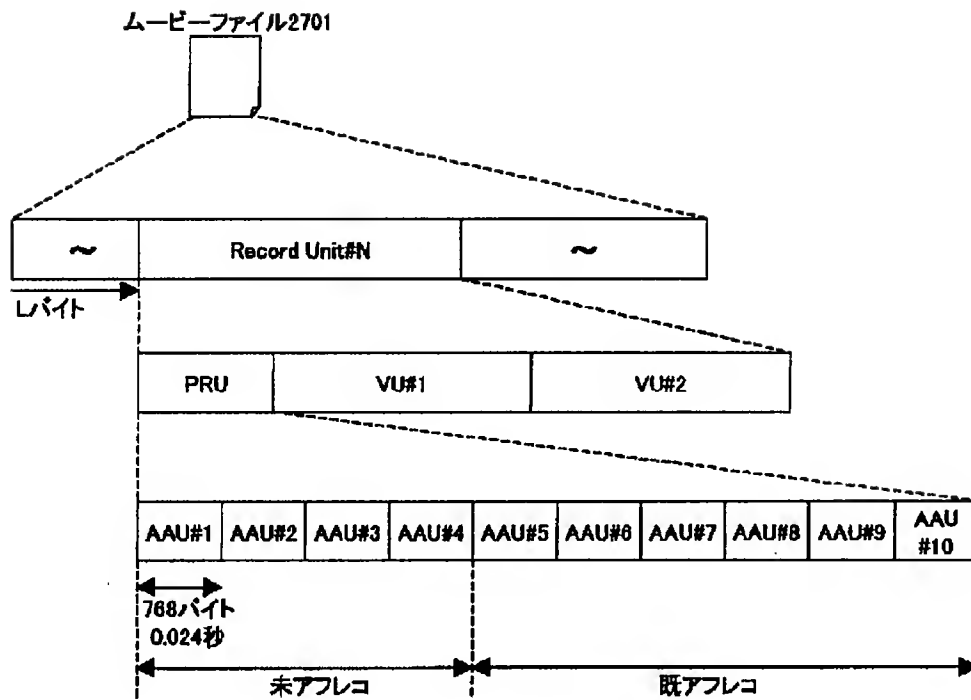
【圖 23】



【圖 28】



【図25】



【図26】

```

Movie atom {
  Movie extends atom {
    Track extends atom /* ビデオトラック */
    Track extends atom /* メインオーディオトラック */
    Track extends atom { /* アフレコ領域トラック */
      default-sample-duration = 2160
      default-sample-size = 768
    }
    Track extends atom { /* アフレコオーディオトラック */
      default-sample-duration = 2160
      default-sample-size = 768
    }
  }
}

```

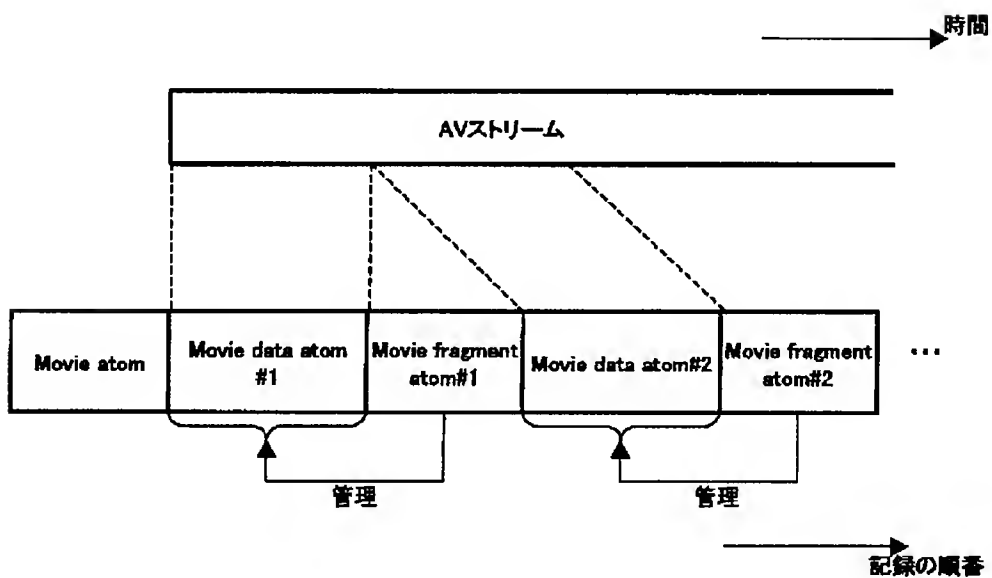
【図27】

```

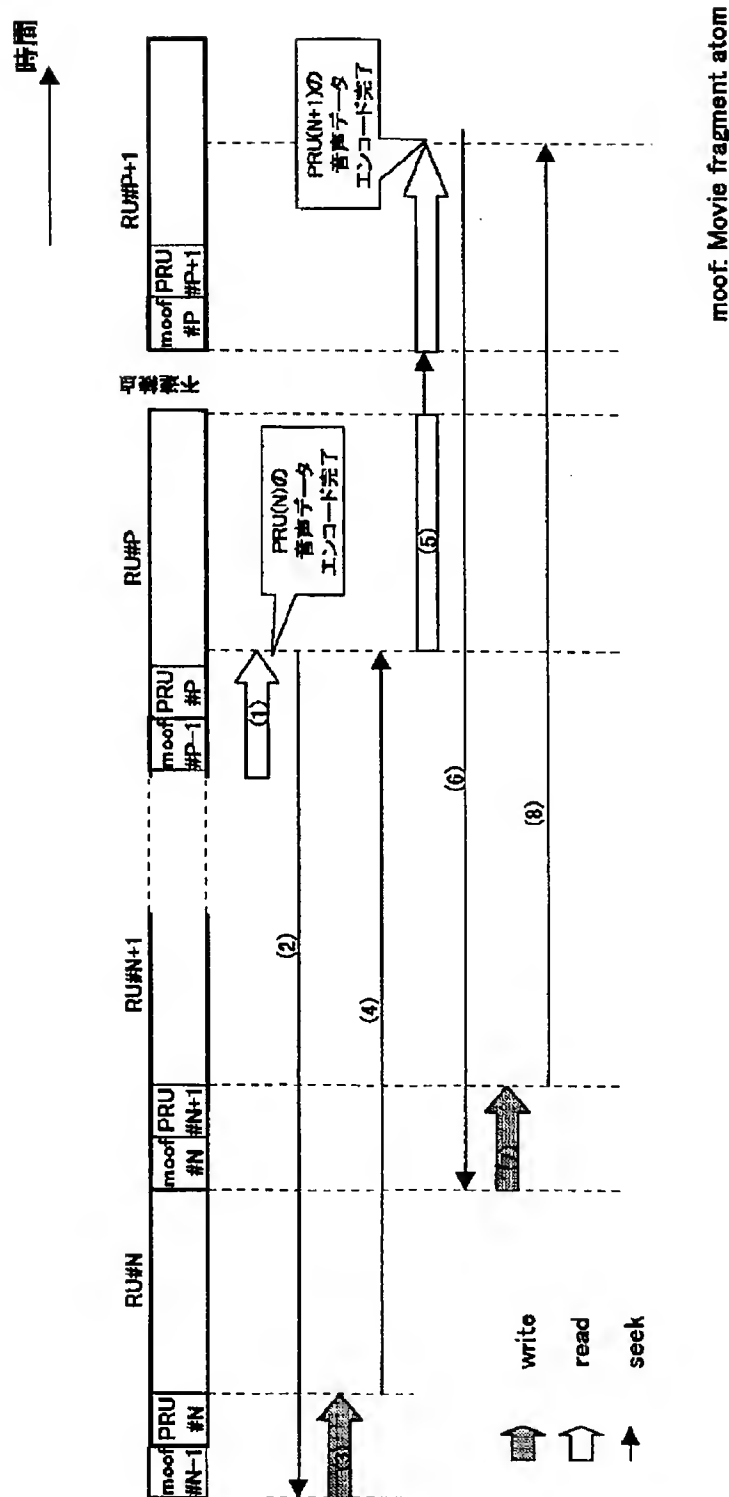
Movie fragment atom {
  Movie fragment header atom
  Track fragment atom /* ビデオトラック */
  Track fragment atom /* メインオーディオトラック */
  Track fragment atom { /* アフレコ領域トラック */
    Track fragment header atom {
      base-data-offset = L
    }
    Track fragment run atom #1 {
      sample-count = 10
      data-offset = 0
    }
  }
  Track fragment atom { /* アフレコオーディオトラック */
    Track fragment header atom {
      base-data-offset = L
    }
    Track fragment run atom #1 {
      sample-count = 6
      data-offset = 4 × 768
    }
  }
}

```

【図32】

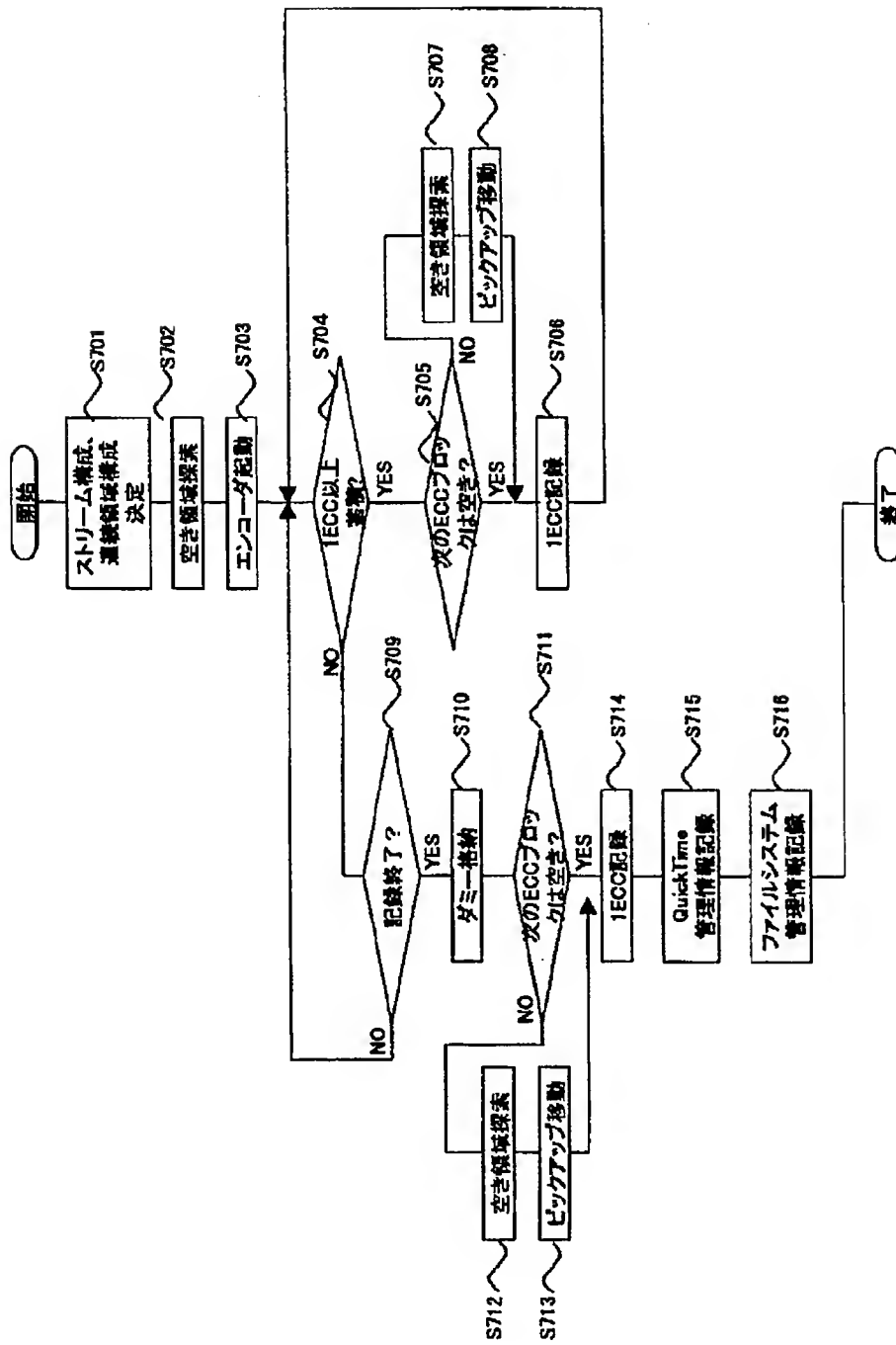


【図29】

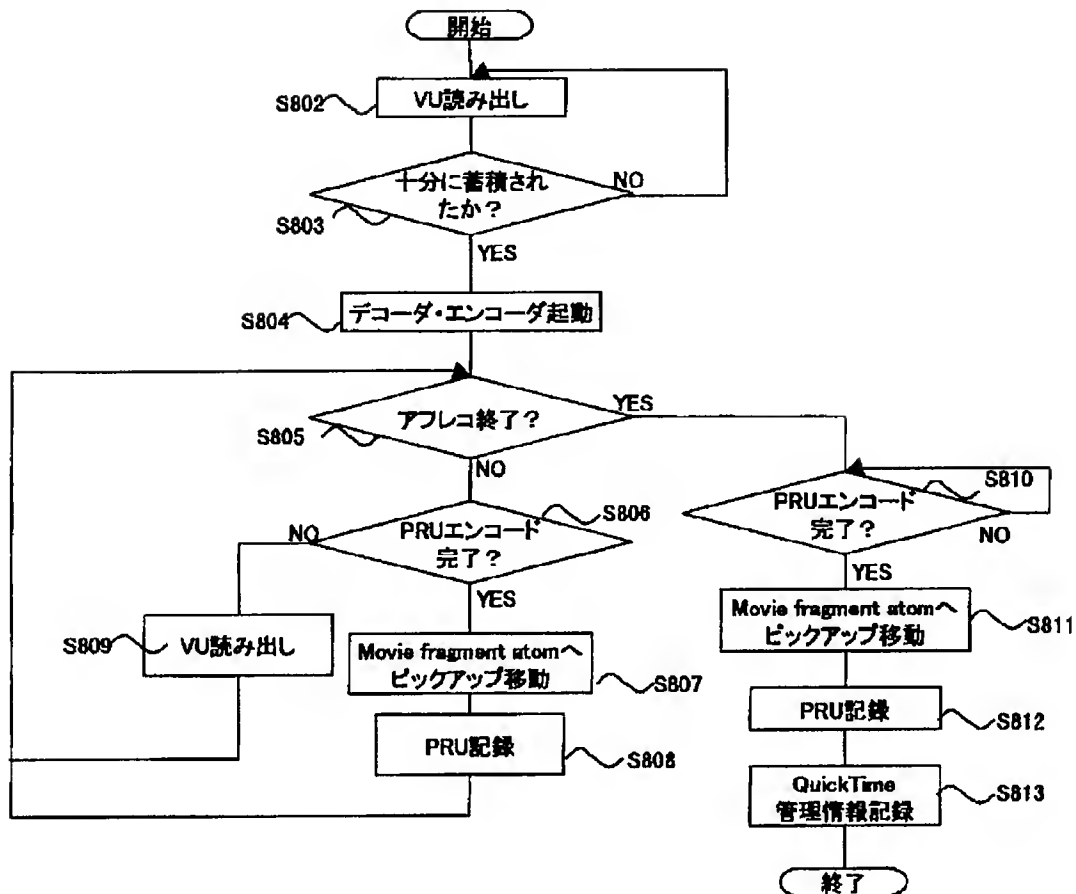




【図30】



【図31】



フロントページの続き

(72)発明者 山口 孝好  
大阪府大阪市阿倍野区長池町22番22号 シ  
ャープ株式会社内

Fターム(参考) SC053 FA23 FA30 GA11 GB05 GB37  
5D044 AB05 AB07 BC01 BC04 CC04  
DE17 DE27 DE48 DE57 EF05  
EF07  
5D110 AA13 AA19 BB06 DA04 DA06  
DA12 DA15 DB02 DD13